AutoNLP

Hugging Face

Mar 08, 2021

GETTING STARTED

AutoNLP: Auto training and fast deployment for state-of-the-art NLP models

AutoNLP is an automatic way to train, evaluate and deploy state-of-the-art NLP models for different tasks. Using AutoNLP, you can leave all the worries of selecting the best model, fine-tuning the model or even deploying the models and focus on the broader picture for your project/business.

CHAPTER ONE

MAIN FEATURES:

- Automatic selection of best models given your data
- Automatic fine-tuning
- Automatic hyperparameter optimization
- Model comparison after training
- Immediate deployment after training
- CLI and Python API available

CHAPTER

TWO

SUPPORTED TASKS

Currently, AutoNLP supports the following tasks:

- Binary classification: one sentence has one target associated with it and there are two unique targets in the dataset
- Multi-class classification: one sentence has one target associated with it and there are more than two unique targets in the dataset
- Entity extraction: also known as named entity recognition or token classification. This task consists of one sentence and in the sentence, each token is associated to a particular label

CHAPTER

THREE

SUPPORTED LANGUAGES

Currently, AutoNLP supports the following languages:

- English: en
- French: fr
- German: de
- Finnish: fi
- Hindi: hi
- Spanish: es
- Chinese: zh
- Dutch: nl

If the language you want to use is not listed, please create an issue here: https://github.com/huggingface/autonlp/issues and we will try our best to add the languages you need.

3.1 Installation

3.2 Binary Classification

Binary classification is one the most popular supervised classification problem one might come across when dealing with NLP problems. AutoNLP makes it super easy to train binary classification models on your data. Let's assume we are training a model for sentiment detection. The dataset has two sentiments: positive & negative.

Let's assume our data is in CSV format and looks something like the following:

Here, we see only three samples but you can have as many samples as you like: 5000, 10000, 100000 or even a million or more!

Once you have the data in the format specified above, you are ready to train models using AutoNLP. Yes, it's that easy.

The first step would be login to AutoNLP:

```
$ autonlp login --api-key YOUR_HUGGING_FACE_API_TOKEN
```

If you do not know your Hugging Face API token, please create an account on huggingface.co and you will find your api key in settings. Please do not share your api key with anyone!

Once you have logged in, you can create a new project:

```
$ autonlp create_project --name sentiment_detection --language en --task binary_
↪classification
```

During creation of project, you can choose the language using "–language" parameter.

The next step is to upload files. Here, column mapping is very important. The columns from original data are mapped to AutoNLP column names. In the data above, the original columns are "sentence" and "label". We do not need more columns for a binary classification problem.

AutoNLP columns for binary classification are:

- text
- target

The original columns, thus, need to be mapped to text and target. This is done in upload command. You also need to tell AutoNLP what kind of split you are uploading: train or valid.

```
autonlp upload --project sentiment_detection --split train \
            --col_mapping sentence:text,label:target \
            --files ~/datasets/train.csv
```

Similarly, upload the validation file:

```
autonlp upload --project sentiment_detection --split valid \
            --col_mapping sentence:text,label:target \
            --files ~/datasets/valid.csv
```

Please note that you can upload multiple files by separating the paths by a comma, however, the column names must be the same in each file.

Once you have uploaded the files successfully, you can start training by using the train command:

```
$ autonlp train --project sentiment_detection
```

And that's it!

Your model will start training and you can monitor the training if you wish.

3.3 Multi Class Classification

Multi-class classification is one the most popular supervised classification problem one might come across when dealing with NLP problems. AutoNLP makes it super easy to train multi-class classification models on your data. Let's assume we are training a model for sentiment detection. The dataset has three sentiments: positive, negative & neutral.

Let's assume our data is in CSV format and looks something like the following:

Here, we see only three samples but you can have as many samples as you like: 5000, 10000, 100000 or even a million or more!

Once you have the data in the format specified above, you are ready to train models using AutoNLP. Yes, it's that easy.

The first step would be login to AutoNLP:

```
$ autonlp login --api-key YOUR_HUGGING_FACE_API_TOKEN
```

If you do not know your Hugging Face API token, please create an account on huggingface.co and you will find your api key in settings. Please do not share your api key with anyone!

Once you have logged in, you can create a new project:

```
$ autonlp create_project --name sentiment_detection --language en --task multi_class_
↪classification
```

During creation of project, you can choose the language using "–language" parameter.

The next step is to upload files. Here, column mapping is very important. The columns from original data are mapped to AutoNLP column names. In the data above, the original columns are "sentence" and "label". We do not need more columns for a multi-class classification problem.

AutoNLP columns for multi-class classification are:

- text
- target

The original columns, thus, need to be mapped to text and target. This is done in upload command. You also need to tell AutoNLP what kind of split you are uploading: train or valid.

```
autonlp upload --project sentiment_detection --split train \
            --col_mapping sentence:text,label:target \
            --files ~/datasets/train.csv
```

Similarly, upload the validation file:

```
autonlp upload --project sentiment_detection --split valid \
            --col_mapping sentence:text,label:target \
            --files ~/datasets/valid.csv
```

Please note that you can upload multiple files by separating the paths by a comma, however, the column names must be the same in each file.

Once you have uploaded the files successfully, you can start training by using the train command:

```
$ autonlp train --project sentiment_detection
```

And that's it!

Your model will start training and you can monitor the training if you wish.

3.4 Entity Extraction